Probabilistic Robotics: Python Simulations of Mobile Robots and Core Robotics Algorithms

UC San Diego
JACOBS SCHOOL OF ENGINEERING
Electrical and Computer Engineering

GEAR
Guided Engineering Apprenticeship in Research

Farnia Nafarifard (B.S. ECE '23)
Advisor/Mentor: Prof. Nikolay Atanasov, Arash Asgharivaskasi
Existential Robotics Laboratory (ERL) at University of California, San Diego (UCSD)

Objective

Autonomy in robotics is an interdisciplinary field that college students don't get to explore in depth unless they have taken a probabilistic math class or graduate courses as undergraduates. By exploring the pillars of this important field, this project hopes to give young undergraduate students the opportunity to learn about the current problems of robotics and the skills necessary to solve them.

Introduction

In robotics, a major interest of engineers is the robots capable of mapping their surrounding environments. This task can be done through various algorithms, probability and sensor measurements. [1] The problem of **simultaneous localization and mapping (SLAM)** has drawn immense attention in the field of mobile robotics. SLAM is the building of an environment's mapping from a sequence of measurements by the moving robot. And as the motion of the robot depends heavily on its understanding of the surrounding environment, robot **localization** is a complement to the **mapping** problem, defining SLAM. A robot's ability to both localize itself and accurately map its environment is an important key prerequisite of truly autonomous robots. [2-5]

Fig 1.1 Physical robot mapping rocks, in a testbed developed for Mars Rover research, using SLAM algos

Analysis

Once the mapping algorithm works, students:

- Have better grasp/understanding of autonomous robots as well as the probabilistic math behind it
- Can implement complex algorithms in Python programming environment
- Are familiar with useful available modules/tools (e.g., numpy, matplotlib, PyBullet, etc)
- Are able to use the outputs of their program to run/ test out their localization algorithm as well

The setup instructions, theoretical lectures, Python implementations/ simulations are available publicly on GitHub for students to use.

Methodology

Occupancy Grid Mapping

To build the map of an environment, we're given::

I. Global position of the robot (i.e. the position relative to the space, not relative to itself)
II. LiDAR (Light Detection and Ranging) scans (i.e., data collected through robots sensors)

Although the space the robot is placed in is continuous, view it as a **discrete space** by making a grid of the space, allowing the robot to see each cell as a space that is either free or occupied space. Two possible ways to go about solving it:

1. Each space (m) having a binary state of 0 or 1
   a. Free space represented by m = 0
   b. Occupied space represented by m = 1
2. Each space (m) represented through the **probability** of whether or not it is occupied -- p(m | z, x)
   a. z - measurements from sensors
   b. x - positions of our robot at different times

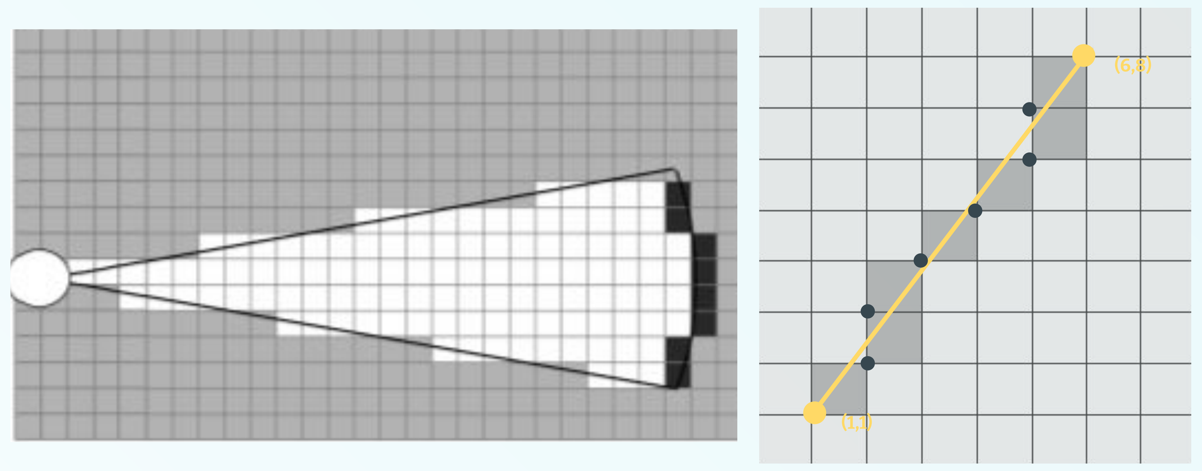

Fig 2.1 Light Detection and Ranging (LiDAR), Bresenham algorithm

Algorithm occupancy_grid_mapping($\{l_{t-1,i}\}, x_t, z_t$):
  for all cells $\mathbf{m}_i$ do
    if $\mathbf{m}_i$ in perceptual field of z_t then
      $l_{t,i} = l_{t-1,i} + \textbf{inverse\_sensor\_model}(\mathbf{m}_i, x_t, z_t) - l_0$
    else
      $l_{t,i} = l_{t-1,i}$
    endif
  endfor
  return $\{l_{t,i}\}$

Fig 2.2 Occupancy grid mapping implementation pseudocode

Localization

Determining where a mobile robot is with respect to its environment, given a map of the robot's environment. We'll assume that we're given:

I. The map of the environment m
II. Series of control commands u
III. Series of observations z

Using the given inputs, we'll determine the robot's position and orientation at every timestep by **particle filter algorithm**

1. Initially, all points in space have same weights, (probability of robot's current position)
2. At every time step:
   a. Predict the current position of the robot
   b. Update the weights from the inputs
   c. Resample particles

Ongoing Work

- **Motion/Path planning**: problem of finding a sequence of valid configurations that moves the object from the source to destination efficiently
- **PyBullet Simulations**: tools and softwares that enable imitation/visualization of the operations of the real-world processes and systems

Conclusion

Having an understanding of the fundamental concepts and core robotics algorithms enables students to have the necessary knowledge to grow in the field, and apply their knowledge/skills to projects/problems in controls & machine learning.

References

[1] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem. Proceedings of the AAAI National Conference on Artificial Intelligence, 2002.
[2] G. Dissanayake, P. Newman, S. Clark, H.F. Durrant-Whyte, and M. Csorba. A solution to the simultaneous localisation and map building (SLAM) problem. IEEE Transactions of Robotics and Automation, 2001.
[3] D. Kortenkamp, R.P. Bonasso, and R. Murphy, editors. AI-based Mobile Robots: Case studies of successful robot systems, MIT Press, 1998.
[4] C. Thorpe and H. Durrant-Whyte. Field robots. ISRR-2001.
[5] J.J. Leonard and H.J.S. Feder. A computationally efficient method for large-scale concurrent mapping and localization. ISRR-99.

Results

Figure 3.1 Simulation/visualization of robot's path/trajectory

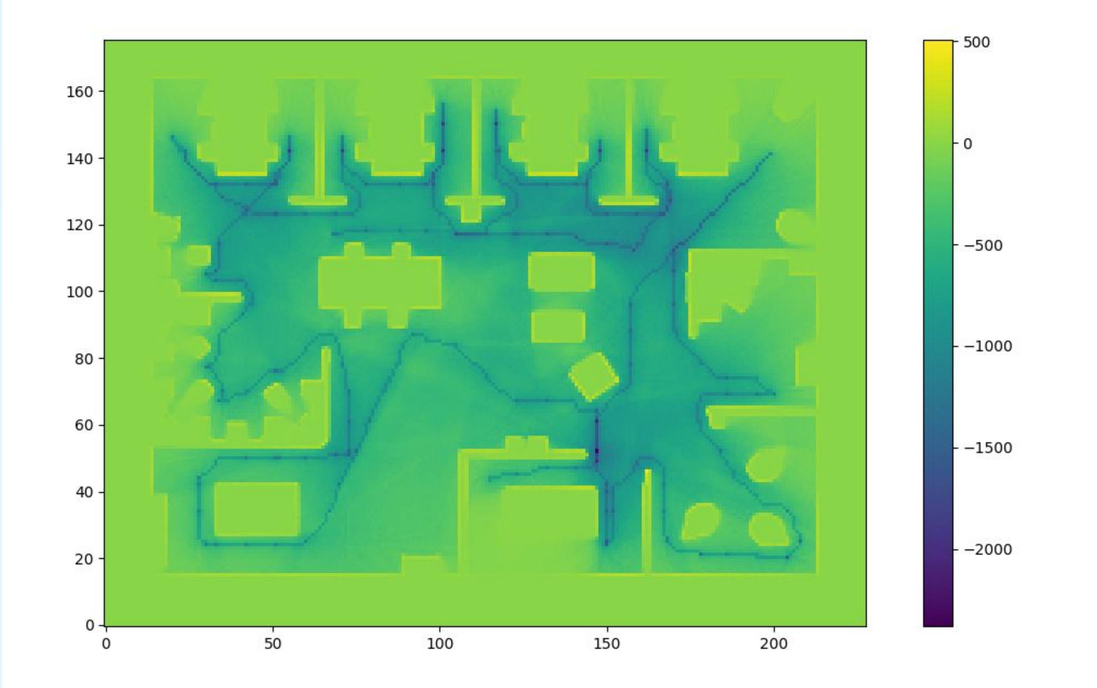

Figure 3.2 Simulation result of log-odds mapping algorithm

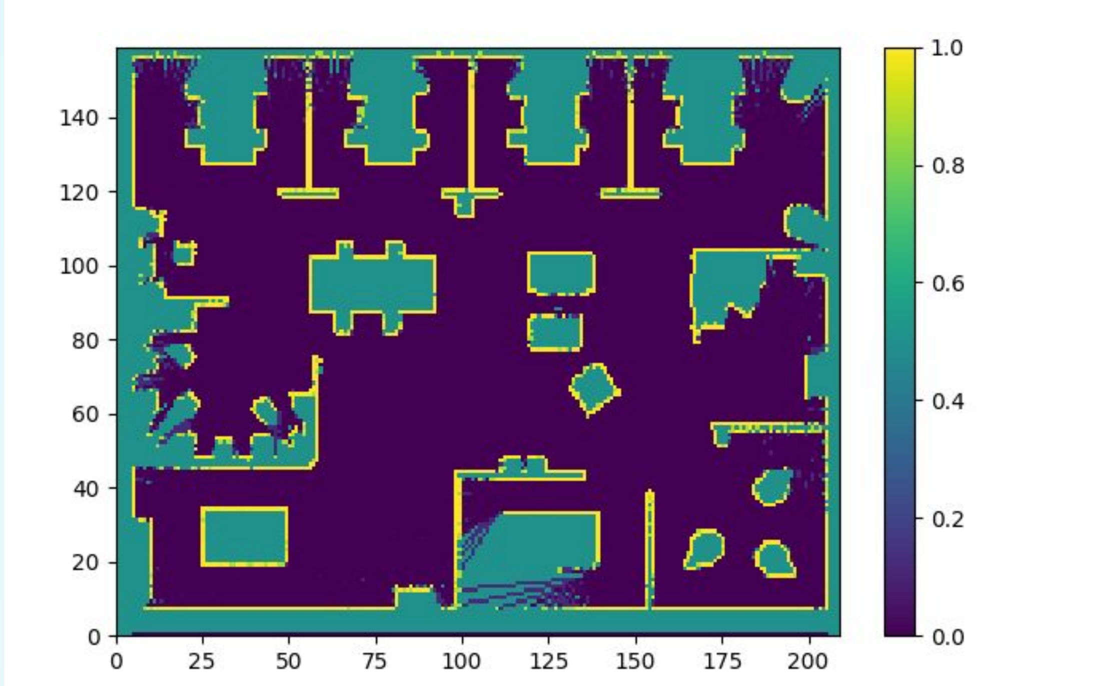

Figure 3.3 Simulation result of log-odds mapping algorithm (on probability scale)